Bitácora detallada: Modernización de la interfaz gráfica

Archivos modificados

- **E:\Admin7\master\frontend\App.css** - Se actualizaron los estilos para darle un aspecto más moderno a la aplicación

Cambios realizados

1. **Encabezado principal**
    - Se reemplazó el color sólido por un degradado moderno (azul)
    - Se mejoró el espaciado y los elementos visuales
    - Se agregó posición sticky para que permanezca visible al desplazarse
2. **Botones de navegación y acción**
    - Se actualizaron los botones con bordes redondeados (estilo pill)
    - Se agregaron efectos de hover con transiciones suaves
    - Se implementaron efectos de elevación con sombras
    - Se agregaron indicadores visuales (ripple effect) para feedback de usuario
3. **Explorador de archivos**
    - Se rediseñó el contenedor principal con bordes más suaves
    - Se agregaron efectos de hover para mejorar la interacción
    - Se implementó un indicador visual con una línea de color en el título
4. **Elementos de archivo y carpeta**
    - Se modernizaron las tarjetas con efectos de elevación
    - Se agregaron indicadores de color para diferenciar archivos y carpetas
    - Se mejoraron las transiciones y efectos visuales al interactuar
    - Se añadieron bordes superiores con códigos de color para mejor identificación
5. **Iconos**
    - Se mejoraron los iconos con fondos circulares sutiles
    - Se añadieron efectos de escala al hacer hover
    - Se aplicaron colores distintivos para carpetas y archivos
6. **Formulario de inicio de sesión**
    - Se rediseñó el contenedor con bordes más suaves y efectos de elevación
    - Se aplicó un degradado sutil al fondo de la página
    - Se mejoró el diseño del botón de inicio con gradiente y efectos
    - Se refinaron los campos de entrada con mejor contraste y transiciones
7. **Panel de gestión de usuarios**
    - Se actualizó el diseño para ser consistente con el nuevo estilo
    - Se mejoró el encabezado con un indicador visual
    - Se refinaron los botones de acción con el mismo estilo global

Pendiente para la próxima sesión

1. **Mejorar la tabla de usuarios**

    - Actualizar el diseño de la tabla para ser más moderna
    - Agregar efectos visuales a las filas y la interacción
2. **Perfeccionar el modal de gestión de usuarios**
    - Mejorar el diseño del contenedor modal
    - Actualizar los botones de cierre y los efectos de transición
3. **Refinamiento de la vista móvil**
    - Revisar y mejorar la respuesta en dispositivos móviles
    - Asegurar que los elementos de interfaz se vean correctamente
4. **Actualizar componentes adicionales**
    - Refinar componentes que no fueron actualizados en esta sesión
    - Asegurar consistencia visual en todo el sitio
5. **Mejorar formularios de búsqueda y filtrado**
    - Actualizar el diseño de los campos de búsqueda
    - Mejorar los controles de filtrado para mayor usabilidad

El trabajo realizado ha transformado significativamente la apariencia de la aplicación, dándole un aspecto más moderno, profesional y consistente. Todos los cambios se han realizado respetando la estructura y funcionalidad original, enfocándose exclusivamente en mejorar el aspecto visual.

Bitácora detallada de la implementación de límites de tamaño personalizados por bucket

Situación inicial

El sistema tenía un único límite global de espacio de almacenamiento (800MB) para todos los buckets, independientemente de cuál se estuviera utilizando. Además, contaba con un límite máximo de tamaño de archivo individual de 36MB configurado en Multer.

Solución implementada

Paso 1: Eliminación de duplicación en userRoleMap

- Se identificó un error de sintaxis en el archivo server.js
- El error estaba causado por una definición duplicada del objeto userRoleMap
- Se eliminaron las líneas duplicadas:
- `// Bucket personal1'adminpersonal1': 'admin','usuariopersonal1': 'user'};`

Paso 2: Creación de mapeo de tamaños de bucket

- Se agregó un nuevo objeto `bucketSizeMap` que define límites personalizados para cada bucket:
- `// Definir tamaños máximos para cada bucket (en MB)const bucketSizeMap = {  'master': 200,            // 200 MB para`

```
master  'contenedor001': 500,     // 500 MB para contenedor001
'contenedor002': 300,     // 300 MB para los demás buckets
contenedor  'contenedor003': 300,  'contenedor004': 300,
'contenedor005': 300,  'contenedor006': 300,  'contenedor007':
300,  'contenedor008': 300,  'contenedor009': 300,
'contenedor010': 300,  'contenedor011': 300,  'contenedor012':
300,  'contenedor013': 300,  'pruebas': 100,         // 100 MB
para pruebas  'personal1': 150         // 150 MB para
personal1};// Tamaño predeterminado para buckets no
especificados (en MB)const defaultBucketMaxSize = 800;
```

Paso 3: Modificación del endpoint `/api/bucket-size`

- Se actualizó este endpoint para:
    1. Verificar que solo los administradores puedan acceder a las estadísticas de almacenamiento
    2. Obtener el límite específico de cada bucket usando `bucketSizeMap`
    3. Calcular correctamente el porcentaje de uso y espacio restante basado en el tamaño específico
    4. Mejorar la respuesta JSON para incluir todos los datos relevantes

Paso 4: Actualización de la validación en `/api/upload`

- Se modificó la validación de tamaño durante la carga de archivos para:
    1. Obtener el límite específico del bucket actual desde `bucketSizeMap`
    2. Convertir el tamaño de MB a bytes para la comparación
    3. Actualizar los mensajes de error para mostrar el límite específico del bucket
    4. Mejorar el registro (logs) para incluir el límite específico

Especificaciones técnicas resultantes

1. **Límites de tamaño por bucket**:
    - bucket "master": 200 MB
    - bucket "contenedor001": 500 MB
    - buckets "contenedor002" a "contenedor013": 300 MB cada uno
    - bucket "pruebas": 100 MB
    - bucket "personal1": 150 MB
    - Cualquier otro bucket no especificado: 800 MB (predeterminado)
2. **Límite de tamaño por archivo individual**: 36 MB (sin cambios)
3. **Comportamiento de la API**:
    - `/api/bucket-size`: ahora muestra estadísticas específicas para cada bucket (solo visible para administradores)
    - `/api/upload`: valida contra el límite específico del bucket en uso
4. **Respuesta JSON de bucket-size**:

```
5. {
6.   "success": true,
7.   "bucket": "[nombre-del-bucket]",
8.   "sizeBytes": 12345678,
9.   "sizeMB": 12.34,
10.    "maxSizeMB": 300,
11.    "percentUsed": 4.11,
12.    "remainingMB": 287.66
13.  }
```

Beneficios de la implementación

1. **Gestión de recursos más eficiente**: Ahora es posible asignar diferentes límites de almacenamiento según las necesidades específicas de cada bucket.
2. **Mejor experiencia de usuario**: Los mensajes de error son más precisos, mostrando el límite específico del bucket en uso.
3. **Control de acceso mejorado**: Solo los administradores pueden ver las estadísticas de almacenamiento.
4. **Escalabilidad**: El sistema permite fácilmente ajustar los límites para buckets específicos modificando el objeto `bucketSizeMap`.
5. **Seguridad**: Se evita que usuarios con buckets más pequeños consuman recursos excesivos.